# 保持细节特征的带纹理模型的高质量简化算法 *

李世俊 a，姜晓彤 a，唐　慧 b†

(东南大学 a. 仪器科学与工程学院; b. 计算机科学与技术学院, 南京 210096)

**摘　要：**目前，三维场景应用越来越多，复杂的模型给实时渲染造成了较大压力，模型简化成为必要的一步。针对大多算法在简化率较大的情况下易丢失模型的细节特征的问题，引入顶点尖锐度的概念，并基于 QEM(quadric error metric)折叠代价给出一种改进的折叠代价，能更多地保留模型的细节特征；同时针对大多简化算法不包含纹理处理的问题，引入纹理变化因子，更多保留纹理的细节部分。在简化的基础上，还提出一种网格局部优化算法，解决简化后网格质量不高的问题。实验结果表明，本算法不仅可以保持模型的细节特征和纹理的完整性，同时得到的模型网格质量高。

**关键词：**三维模型；网格简化；顶点尖锐度；二次误差；局部优化

**中图分类号：**TP391　　doi: 10.19734/j.issn.1001-3695.2018.04.0481

# High-quality simplified algorithm of texture model for detailed features preserving

Li Shijun[a], Jiang Xiaotong[a], Tang Hui[b†]

(*a. School of Instrument Science & Engineering*, *b. School of Computer Science & Technology*, *Southeast University*, *Nanjing* 210096, *China*)

**Abstract:** At present, there are an increasing number of applications of 3D scene, and the complicated models put great pressure on real-time rendering. The simplification of the models becomes a necessary step. In order to solve the problem that most algorithms are prone to lose the detailed features of the models in the case of large simplification rate, this paper introduced the concept of vertex sharpness and developed an improved folding cost based on QEM(Quadric Error Metric) cost, which can preserve more detailed features of the model. At the same time, aiming at the problem that most of the simplified algorithms do not contain texture processing, this paper introduced the texture variation factor to preserve the details of the texture. On the basis of simplification, this paper proposed a mesh local optimization algorithm to solve the problem of low quality of mesh after simplification. The experimental results show that the proposed algorithm can not only preserve the details of the model and the integrity of texture, but also obtain the model with high-quality mesh.

**Key words:** 3D model; Mesh simplification; Vertex sharpness; Quadric Error Metric; Local optimization



## 0　引言

目前，互联网家居日益兴起，三维场景的应用越来越多。在室内设计行业，往往可以通过扫描真实物体后进行三维重建，进而输出其三维模型。但是该方法得到的模型往往十分精细，所含三角面片数量过多，不利于在移动端的实时渲染，于是模型简化显得尤为必要。

网格是三维模型的基本组成单元，简化模型即是对网格的简化，其中三角网格是表达三维模型最常用的形式。网格简化算法研究较多的大致可以分为两种类型:顶点重采样型和元素删除型。顶点重采样是将模型划分成很多区域，对每块区域内部顶点进行剔除，然后布入新的顶点，再进行三角剖分，通过控制重采样点的数量可以控制模型的简化率。而元素删除型是通过逐步地删除三角网格的元素来达到简化的目的，根据删除对象的不同，可以将其分为顶点元素删除法[1]、三角形收缩法[2,3]和边折叠法[4,5]。其中，边折叠由于简化速度快，简化效率高，同时可以保持原模型的拓扑结构，已经成为目前应用最广泛的简化算法。

Hoppe[6]最先提出可以使用能量函数来指导模型的简化，并给出对三维扫描仪测量的数据模型进行简化的实例，效果很好，但效率较低。Garland 等人[7]提出可以使用二次误差来度量边折叠的代价，该算法简化速度快，但是简化太过均匀，容易丢失细节特征。Ozaki 等人[8]基于 QEM 算法，使用基于机器学习方法的线性分类器对网格进行划分，实现了对特大模型的简化，然而细节特征依旧容易丢失。针对细节特征丢失问题，Ho 等人[9]提出一种可控的网格简化算法，用户可以选择需要的区域保

**收稿日期：**2018-04-16；**修回日期：**2018-05-24　　**基金项目：**江苏省自然科学基金青年基金资助项目（BK20150647）

**作者简介：**李世俊（1994-），男，安徽舒城人，硕士研究生，主要研究方向为图形图像处理；姜晓彤（1975-），男，副教授，博士，主要研究方向为虚拟现实、人工智能；唐慧（1981-），女（通信作者），讲师，博士，主要研究方向为图形图像处理（corinna@seu.edu.cn）.

留细节特征，但此方法较为繁琐，对操作者要求较高。黄佳等人[10]提出一种渐进网格简化算法，通过一环和二环三角面的法矢变化作为折叠代价，对细节特性保持较好。但由于仅考虑对细节的保留，该算法对平坦区域的简化顺序没有很好的约束。王竣等人[11]提出一种基于边曲率的简化算法，通过将边曲率加入到二次误差来增加尖锐部分的折叠代价，但对边曲率的估计太过简单。钱勋波等人[12]从局部三角形的不平度入手，通过三角形的折叠来完成模型的简化，然而由于三角形折叠易丢失特征的先天缺陷，对模型细节特性的保留没有边折叠效果好。刘峻等人[13]通过顶点曲率的引入，来增强算法对特征区域的保持，效果较好，但是由于仅考虑顶点与一环邻域点的最大法矢夹角与总面积，没有考虑引起该最大夹角的三角面的面积大小，容易引起判断失误。同时，以上所提算法均没有考虑纹理的变化，然而在实际应用中，纹理的保持必不可少。

为了解决上述问题，本文引入顶点尖锐度的概念，并以此改进 QEM 算法中顶点的 Q 值，可以准确地判断出模型的尖锐顶点，增大其折叠代价，同时对模型的平坦部分也有很好的约束。同时，本文针对纹理的变化引入纹理因子，能在不改变算法复杂度的情况下最大限度地保持纹理的完整性。此外，本文还提出一种网格局部优化算法，解决由于不均匀简化导致的网格质量偏低的问题。实验结果表明，本文算法在简化时可以很好地保持模型的细节特征和纹理的完整性，同时简化后的模型网格均匀，视觉效果较好。

## 1　基于边折叠的网格简化算法

### 1.1　边折叠

图 1 给出了边折叠的示意图。边折叠算法是将三角网格所有的边根据删除代价升序排序，每次操作取出队列最顶端的一条边（即折叠代价最小的），如图 1 所示的边 $v_1v_2$，将该边的两个端点 $v_1,v_2$ 合并到某一个误差最小的新顶点 $v_0$，删除顶点 $v_1,v_2$，添加新点 $v_0$，删除同时拥有这两个顶点的两个三角形 $v_1v_2v_3$ 和 $v_1v_2v_4$，删除相应的边 $v_1v_2,v_1v_3,v_2v_3,v_1v_4,v_2v_4$，添加新生成的边 $v_0v_3,v_0v_4$，并同时更新顶点周围顶点信息，边信息等拓扑结构。重复以上步骤直至剩余三角形数目达到要求，这就是边折叠的基本过程。

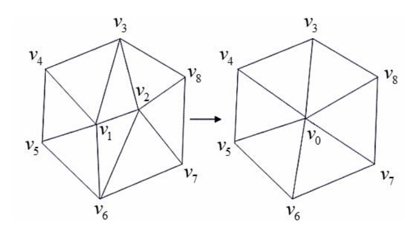



图 1　边折叠算法示意图

Fig.1　Schematic diagram of Edge folding

### 1.2　边折叠代价的计算

QEM 算法是 Garland 等人在 1997 年提出的一种基于二次误差度量的边折叠三角网格模型简化算法，该算法简化速度快，简化率高。QEM 算法边折叠的代价通过二次误差来衡量。误差 $\Delta(v_i \to v_0)$ 表示选取的新顶点 $v_0$ 到 $v_i$ 周围所有三角形距离的平方和：

$$\Delta(v_i \to v_0)=\Delta([x_0,y_0,z_0,1]^T) = \sum_{p\in planes(v_i)} (p^T v_0)^2$$

$$= \sum_{p\in planes(v_i)} (v_0{}^T p)(p^T v_0) = v_0{}^T (\sum_{p\in planes(v_i)} K_p) v_0 \quad (1)$$

其中：$planes(v_i)$ 表示 $v_i$ 周围的所有相邻三角形，$p=(a,b,c,d)^T$ 表示由式子 $ax+by+cz+d=0$ 定义的平面，且该方程为归一化的平面方程，即 $a^2+b^2+c^2=1$。故

$$K_p = pp^T = \begin{bmatrix} a^2 & ab & ac & ad \\ ab & b^2 & bc & bd \\ ac & bc & c^2 & cd \\ ad & bd & cd & d^2 \end{bmatrix} \quad (2)$$

记 $Q(v_i)=\sum_{p\in planes(v_i)} K_p$，则 $\Delta(v_i \to v_0)=v_0{}^T Q v_0$ 为其中一个顶点的二次误差。为了简单起见，边折叠过程的二次误差可以定义为两个顶点的二次误差之和，即

$$\begin{aligned} \varepsilon_{12} &= \Delta(v_1 \to v_0)+\Delta(v_2 \to v_0) \\ &= v_0{}^T Q(v_1) v_0 + v_0{}^T Q(v_2) v_0 \\ &= v_0{}^T (Q(v_1)+Q(v_2)) v_0 \end{aligned} \quad (3)$$

记 $Q=Q(v_1)+Q(v_2)$ 为误差矩阵，则每条边的折叠代价为

$$\Delta M = v_0 Q v_0{}^T \quad (4)$$

将每条边的三个候选点(左端点、右端点和中点)的 $\Delta M$ 求出，选择 $\Delta M$ 最小的点为待折叠的选取点，并记录此时的边折叠代价 $\Delta M$。

## 2　改进的简化代价

### 2.1　基本概念

顶点 V 的一环邻域指的是所有包含该顶点的三角形集合，如图 2 所示所有三角形为顶点 V0 的一环邻域。

边界顶点和复杂顶点定义如下：统计该顶点所有相邻三角形所包含的各顶点的出现次数，如果所有相邻顶点在该顶点的相邻三角形中均只出现 2 次，则该顶点为普通顶点（如图 3 中顶点 *A*）；如果某相邻顶点在该顶点的所有相邻三角形中只出现了 1 次，则对应的相邻顶点和该顶点均为边界点（如图 3 中的顶点 *B* 和 *C*）；如相邻顶点在该顶点的相邻三角形中出现了大于 2 次（3 次），则对应的相邻顶点和该顶点均为复杂顶点（如图 3 中的顶点 *D* 和 *E*）；如果该顶点没有相邻三角形，则该顶点为孤立顶点（如图 3 中的顶点 *F*）。

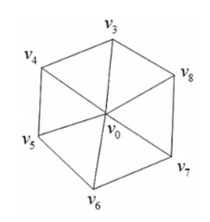



图 2　V0 的一环邻域

Fig.2　1-ring neighborhood

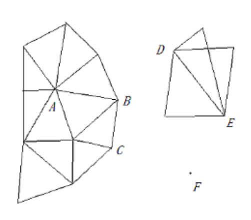



图 3　边界顶点和复杂顶点

Fig.3　Boundary vertices and complex vertices

本文常用符号解释：

$v_t$ 表示模型网格的第 t 个顶点；

$S_{v_t,i}$ 表示顶点 $v_t$ 一环邻域第 i 个三角面的面积；

$N_{v_t,i}$ 表示顶点 $v_t$ 一环邻域第 i 个三角面的法向量；

$S_{texv_t,i}$ 表示顶点 $v_t$ 一环邻域第 i 个三角面对应的纹理面积；

$S_{v_t}$ 表示顶点 $v_t$ 一环邻域所有三角面的面积和；

$<\vec{a},\vec{b}>$ 表示向量 $\vec{a}$ 与向量 $\vec{b}$ 的夹角。

### 2.2　顶点尖锐度的计算

在细节特征较多的模型中，如果仅采用二次误差度量，尖锐部分极易丢失，因此本文引入顶点尖锐度这一概念，并将其融合入顶点矩阵中，在简化率较大的情况下，很好的保持了这些细节特征。

如图 4 所示，通常在顶点较尖锐的部分，三角面之间的法矢夹角偏大，而在较平坦区域三角面之间的法矢夹角相对较小，当两个面绝对平坦时，其法矢夹角显然为零。

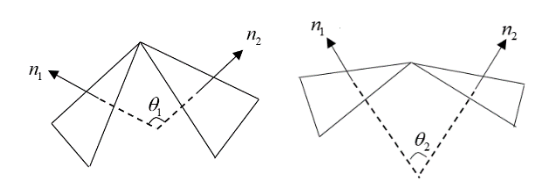



图 4　三角面之间的法矢夹角示意图

Fig.4　Normal vector angle between triangular faces

所以，顶点尖锐度 $\sigma$ 可由其一环邻域所有三角面法矢间的夹角来反映，其计算公式如下：

$$\sigma(v_t)=\frac{\sum_{i=0}^{n-1}\frac{(S_{v_t,i}+S_{v_t,i+1})}{2S_{v_t}}\cdot<N_{v_t,i},N_{v_t,i+1}>+\frac{(S_{v_t,0}+S_{v_t,n})}{2S_{v_t}}\cdot<N_{v_t,0},N_{v_t,n}>}{n} \tag{5}$$

其中：$n$ 为顶点 $v_t$ 一环邻域内三角面的个数。在式（5）中，对顶点 $v_t$ 一环邻域的三角面依次求其法矢夹角，并取其关于面积的加权平均值，可以有效地反映该区域的尖锐度。其中面积的加入是因为，有时候虽然三角面之间的法矢夹角很大，但由于其三角面本身面积很小，简化时损失很小，需要适当减轻该夹角的比重，更准确地反映该区域的尖锐度。

### 2.3　纹理因子

目前，大多数网格简化算法中，均没有考虑纹理的变化，然而在实际应用中，模型的纹理必不可少，是视觉体验的重要组成部分。本文在研究纹理的重要性后给出以下纹理因子公式：

$$R(v_t)=\frac{\sum_{i=0}^{n}\frac{S_{texv_t,i}}{S_{v_t,i}}}{n} \tag{6}$$

其中：$n$ 为顶点 $v_t$ 一环邻域内三角面的个数。在式(6)中，考虑到如果某个三角面纹理面积占几何面积的比重大，则该区域的简化将带来较多的纹理损失，因此该公式可表达该区域纹理的重要程度。同时，在简化过程中，纹理坐标将保持与顶点坐标相同的更新操作。

### 2.4　改进的边折叠代价

在 QEM 边折叠代价的基础上，本文给出了改进的边折叠代价函数，将顶点的尖锐度和纹理变化因子综合嵌入顶点的 $Q$ 矩阵中，避免了简化时再进行计算，减少了算法的复杂度。改进的边折叠代价如下：

$$\varepsilon_{12}=v_0^T(A^{\sigma(v_1)+R(v_1)}Q(v_1)+A^{\sigma(v_2)+R(v_2)}Q(v_2))v_0 \tag{7}$$

其中：$A$ 为调节因子，可取大于零的实数，可避免 $\sigma+R$ 值过小引起代价的区分度降低，本文取值为 2。

## 3　网格局部优化

由于边折叠算法自身的局限性，尽管在改进的简化算法中已经很好地保持了模型的细节特征，但是网格质量普遍不高，容易出现狭长三角形，而过多的狭长三角形容易造成视觉差异，也给后续的模型处理带来麻烦。因此，本文在研究文献[14,15]的基础上提出一种局部网格优化算法，可以减少低质量网格的数量，达到优化的目的。

### 3.1　网格质量判定标准

网格质量通常由三角形的正则度[16]来判定，即该三角形接近正三角形的程度。某三角形的正则度可以用以下公式来估计：

$$r=\frac{4\sqrt{3}A}{l_1^2+l_2^2+l_3^2} \tag{8}$$

其中：$l_1$, $l_2$, $l_3$ 分别为三角形的三条边的边长，$A$ 为三角形的面积。$r$ 表示三角形的正则度，当 $r=1$ 时，三角形为正三角形；而当三角形十分狭长时，$r\approx 0$。随后进行网格优化时，本文均以此式作为判定依据，并规定 $r<0.5$ 的三角形为狭长三角形。

### 3.2　优化算法

全局优化会丢失一些模型的细节特征，因此本文仅对模型进行局部优化，达到基本消除狭长三角形的目的。

#### 3.2.1 平面网格优化

文献[14]研究了 Voronoi 多边形的一些应用，其可以应用于对平面网格的优化。如图 5 所示，优化步骤为：

a)首选找出顶点 $V$ 一环邻域所有的三角形；

b) 得到关于顶点 $V$ 的 Voronoi 多边形；

c) 将顶点 $V$ 移至 Voronoi 多边形的质心，完成局部优化。

由于 Voronoi 多边形构造复杂，在要求不是非常高的情况下，可以简单的取顶点 $V$ 一环邻域多边形的形心作为优化点。

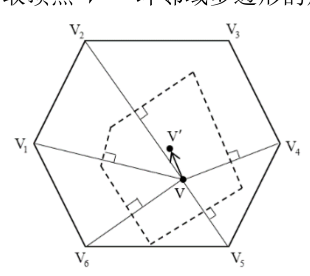



图 5　平面网格优化示意图

Fig.5 Optimization of plane mesh

3.2.2 三维网格的展平

由于 3.2.1 节所提算法仅可用于对平面网格的优化，而本文处理的是三维网格，因此需要对三维网格进行平面化。文献[15]提出一种最小二乘法逼近准保角映射的曲面参数化算法（以下简称 LSCM 算法），可以很好地将复杂网格展平，该算法稳定有唯一解，因此本文选取该算法对网格需要优化的局部进行平面化。如图 6 所示，保角映射 $X$ 将参数域 $(u,v)$ 映射到一个曲面域 X(u,v)，其中对于每个参数 $(u,v)$，其映射 $X$(u,v) 的两条曲线 iso-u 和 iso-v 的切向量相互垂直且长度相等。可以简单理解成，保角映射就是将参数域 $(u,v)$ 的单位圆映射到曲面域 X(u,v) 对应点的切平面的单位圆。

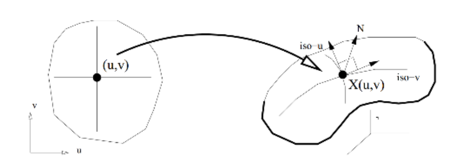



图 6 保角映射示意图

Fig.6 Conformal mapping

三维网格展平需要原曲面尽量在同一平面，这样后续恢复时才不会造成较大误差，因此在对顶点的一环邻域展平时，需要考虑该区域的尖锐度，在较为平坦时才能够进行优化操作，本文通过顶点尖锐度作为依据，仅对尖锐度小于 0.3 的区域进行优化操作。

3.2.3 三维网格的恢复

文献[17]中提出一种网格重构算法，其中对平面网格顶点恢复的算法可用于本文对三维网格的恢复。其描述如下：

对于原始边界顶点，可以采用其原来的三维坐标。而网格内部新的顶点 $u$ 的三维坐标求解方法如下:

先找出三个原始边界顶点的三维坐标 $v_1,v_2,v_3$，其对应的平面坐标为 $u_1,u_2,u_3$，并且点 $u$ 在这三个顶点组成的三角形的内部。可令

$$k_1=\frac{A(u,u_2,u_3)}{A(u_1,u_2,u_3)}\quad k_2=\frac{A(u,u_1,u_3)}{A(u_1,u_2,u_3)}\quad k_3=\frac{A(u,u_1,u_2)}{A(u_1,u_2,u_3)}$$

其中：$A(u_1,u_2,u_3)$ 表示以为 $u_1,u_2,u_3$ 顶点的三角形的面积。则 $u$ 对应三维坐标为

$$v=k_1v_1+k_2v_2+k_3v_3 \tag{9}$$

为了保证恢复三维坐标的精确性，本文取该顶点一环邻域多边形所有满足点 $u$ 在其内部的三角形，分别根据式(9)求其三维坐标，最后求平均值，更精确地还原原网格。

### 3.3 本文算法流程

综上所述，本文算法具体描述如下：

a) 初始化：首先标记模型的拓扑关系，标记每个顶点是否为对边界顶点或复杂顶点。如果一条边中有一个顶点为边界顶点或者复杂顶点，则将其收缩代价函数设为最大值；对所有可收缩边，分别求其收缩至两端点和中点的收缩代价并取最小值 $\varepsilon_{12}$，记录收缩点，并把该代价函数值插入一个优先队列中。所有代价函数计算完毕后，将得到一个按照边的收缩的代价函数从小到大排列的队列 $Q$。

b) 简化过程：执行处于 $Q$ 队列最顶端的操作，更新该操作所影响到的所有的边的收缩代价函数，重新计算更新点的坐标和纹理坐标，更新模型的拓扑关系，同时把修改过的边根据其代价函数重新插入优先队列。

c) 重复步骤 b）直到用于表达模型的三角形数目达到预定要求。

d）遍历简化后网格的顶点，如果顶点尖锐度小于 0.3 判断其一环邻域内有无狭长三角形，如果有，则利用 LSCM 算法将该区域展平，再利用 3.2.1 节所提算法优化。

e）利用 3.2.3 节算法得出新顶点的三维坐标。

f）直至所有顶点优化完毕，结束算法。

## 4 实验结果与分析

为了验证本文算法的有效性，采用 Microsoft Visual Studio 2013 和 OpenGL 编程，在 2.5 GHz Intel Core i5-3230M CPU、5 GB 内存计算机上进行实验研究和分析。



(a)原始模型

(a) Original model



(b)未加入顶点尖锐度

(b)Result without vertex sharpness



(c)未加入纹理因子

(c) Result without texture factor



(d)本文算法

(d) Algorithm in this paper

图 7 人物模型简化 60%示意图

Fig.7 60% simplification of character model

图 7 展示了顶点尖锐度和纹理因子加入对游戏人物模型简化的影响。图 7 中，(a)为人物原始模型，该模型拥有 4019 个三角面；(b)为未加入顶点尖锐度的简化 60%结果，可以看见圈中的刀模型丢失严重；(c)为未加入纹理因子约束的简化 60%结果，圈中的眼部纹理发生明显变形；(d)为使用本文折叠代价函数简化 60%结果，细节特征和纹理保持较好，视觉观感与原模

型最为接近。




(a)原始模型

(a) Original model



(b)未加入顶点尖锐度

(b) Result without vertex sharpness



(c)未加入纹理因子

(c) Result without texture factor



(d)本文算法

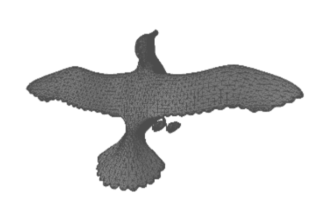
(d) Algorithm in this paper

图 8 僵尸模型简化 65%示意图

Fig.8 65% simplification of corpse model

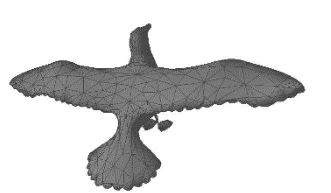
图 8 展示的是顶点尖锐度和纹理因子对僵尸模型简化的影响。图 8(a)是僵尸原始模型，该模型拥有 12686 个三角面。图 8(b)和图 8(c)分别为模型未加入顶点尖锐度和纹理因子的简化结果，可见圈中的领带纹理出现不同程度的错乱。使用本文算法的结果如图 8(d)所示，领带纹理保持与原模型的高度一致性。以上两组实验验证了本文折叠代价在保持细节特征和重要纹理方面的有效性。

为了更加直观地展示本文简化算法的优缺点，本文将从简化模型的几何误差、网格平均质量以及算法耗时三个方面，选取三个网格数量较多的模型与 QEM 算法和刘峻算法进行对比实验，其中几何误差使用 metro[18]中的均方根误差(RMS)来衡量，网格质量使用式(8)估计。



(a)鸽子原模型(16960 个三角形) (b)QEM 简化 80%结果

(a)Original model of pigeon (b) 80% simplified by QEM

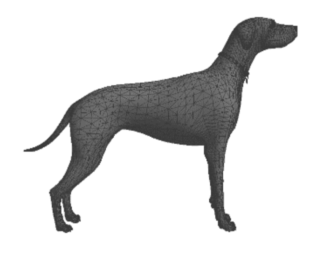


(c) 刘峻算法简化 80%结果 (d)本文算法简化 80%结果

(c) 80% simplified by Liu Jun (d) 80% simplified by this paper

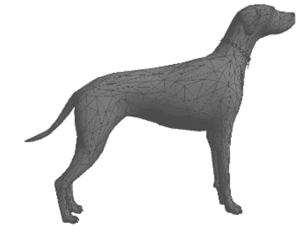
图 9 鸽子模型简化 80%结果

Fig.9 80% simplification of pigeon model



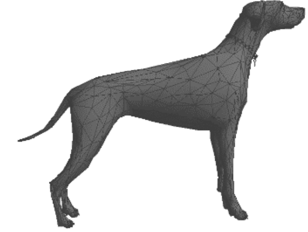
(a)小狗原模型(21293 个三角形)

(a)Original model of dog



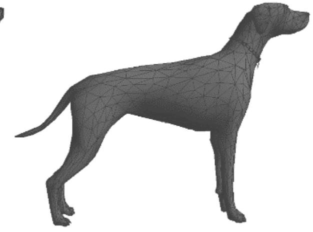
(b)QEM 算法简化 85%结果

(b) 85% simplified by QEM



(c)刘峻算法简化 85%结果

(c) 85% simplified by Liu Jun

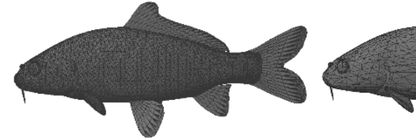


(d)本文算法简化 85%结果

(d) 85% simplified by this paper

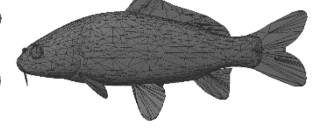
图 10 小狗模型简化 85%结果

Fig.10 85% simplification of dog model



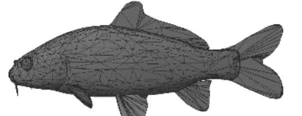
(a)鱼原模型(29188 个三角形)

(a)Original model of fish



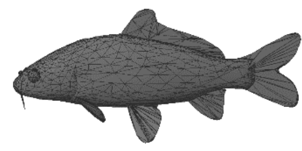
(b)QEM 简化 85%结果

(b) 85% simplified by QEM



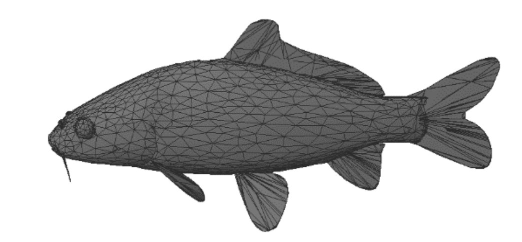
(c)本文算法未局部优化结果

(c) result without local optimization by this paper



(d)刘峻算法简化 85%结果

(d) 85% simplified by Liu Jun



(e)本文算法简化 85%结果

(e) 85% simplified by this paper

图 11 鱼模型简化 85%结果

Fig.11 85% simplification of fish model

图 9 展示的是鸽子模型简化 80%的结果，图 10 是小狗模型简化 85%的结果，图 11 是鱼模型简化 85%的结果。通过以上三组对比实验，从视觉上可以看出本文算法和刘峻算法能得到比 QEM 算法质量更高的网格简化结果。下面将从量化结果

上分析三种方法的优缺点。

表 1　各简化方法的误差、网格质量以及简化时间对比

Table1 Comparison of error, mesh quality and simplified time between this method and other approaches

| 模型 | 方法 | 网格质量(式(8)) | 几何误差(RMS)/$10^{-2}$ | 耗时/ms |
|---|---|---|---|---|
| | QEM 方法 | 0.5918 | **0. 0347** | **187** |
| 鸽子 | 刘峻方法 | 0.6123 | 0.0812 | 368 |
| | 本文方法 | **0.6531** | 0.0848 | 391 |
| | QEM 方法 | 0.5932 | **0.0893** | **281** |
| 小狗 | 刘峻方法 | 0.6038 | 0.2068 | 326 |
| | 本文方法 | **0.6234** | 0.1926 | 313 |
| | QEM 方法 | 0.4695 | **0.0281** | **375** |
| 鱼 | 刘峻方法 | 0.4867 | 0.0693 | 406 |
| | 本文方法 | **0.5092** | 0.0627 | 438 |

通过表 1 的数据可知，在网格质量方面，本文方法得到的网格质量最好，刘峻方法次之，而 QEM 算法得到的网格质量最低，这说明 QEM 方法会产生了较多的狭长三角形，同时本文对优化点的选取方法要优于刘峻方法。通过比较几何误差，可以发现三种算法的误差都很小，均接近于原始模型。其中，QEM 算法误差最小，本文方法和刘峻方法基本相当，略大于 QEM 算法。这是因为，在局部优化环节，本文方法和刘峻方法均对局部顶点做了移动，产生了相对较大的误差。但是在牺牲很小一部分误差的前提下得到更高质量的网格是值得的。

另一方面，本文和刘峻方法均增加了对网格进行局部优化的步骤，因此算法总耗时略大于 QEM 算法，但是三种算法耗时均在毫秒级别，差别可以忽略不计。

另外，由图 11(b)和 11(c)的对比可以发现一个细节。由于本文算法的不均匀简化，在不经过局部优化的简化结果与 QEM 算法对比，出现了更多的狭长三角形，因此网格局部优化对本文算法有着重要的意义。

## 5　结束语

本文在 QEM 简化算法的基础上研究了一种新的网格简化算法，将顶点尖锐度和纹理因子综合融入边折叠代价函数，实验证明该折叠代价不仅可以保留模型细节特征，也可以保留一些重要的纹理部分。此外，在简化后，本文结合曲面参数化和 Voronoi 多边形的应用提出了一种网格局部优化算法，实验证明该算法可有效提高模型的网格质量，提升视觉感受。但是该算法在执行时，需要在顶点尖锐度较低的局部区域进行优化，下一步的工作是研究如何对这些特征较明显的区域进行优化。

## 参考文献：


[1] 罗鹍, 黄魁东, 连明明. 基于顶点删除的三角网格模型简化新方法 [J]. 微电子学与计算机, 2009, 26 (5): 142-143. (Luo Kun, Huang Kuidong, Lian Mingming. New method of triangular mesh simplification based on vertex culling [J]. Microelectronics & Computer, 2009, 26 (5): 142-143. )

[2] 段黎明, 邵辉, 李中明, 等. 高效率的三角网格模型保持特征简化方法 [J]. 光学精密工程, 2017, 25 (2): 460-468. (Duan Liming, Shao Hui, Li Zhongming, etal. Simplification method for feature preserving of efficient triangular mesh model [J]. Optics and Precision Engineering, 2017, 25 (2): 460-468. )

[3] 闫涛. 基于高斯曲率的三角网格模型简化的研究 [J]. 计算机工程与科学, 2012, 34 (12): 126-129. (Yan Tao. Triangular mesh simplification based on Gauss curvature [J]. Computer Engineering & Science, 2012, 34 (12): 126-129. )

[4] 王晓哲, 刘永继, 赵龙波, 等. 基于特征保持的半自动化动态网格简化方法 [J]. 计算机应用研究, 2015 (9): 2839-2843. (Wang Xiaozhe, Liu Yongji, Zhao Longbo, etal. Semi-automated and dynamic mesh simplification algorithm based on feature preserving [J]. Application Research of Computers, 2015 (9): 2839-2843. )

[5] Tang H, Shu H Z, Dillenseger J L, *et al.* Moment-based metrics for mesh simplification [J]. Computers & Graphics, 2007, 31 (5): 710-718.

[6] Hoppe H. Progressive meshes [C]// Proc of SIGGRAPH. New York: ACM Press, 1996: 99-108.

[7] Garland M, Heckbert P. Surface Simplification Using Quadric Error Metric [C]// Proc of the 24th Annual Conference on Computer Graphics and Interactive Techniques. New York: ACM Press, 1997: 209-216.

[8] Ozaki H, Kyota F, Kanai T. Out-of-core framework for QEM-based mesh simplification [C]// Proc of Eurographics Symposium on Parallel Graphics and Visualization. Eurographics Association, 2015: 87-96.

[9] Ho T C, Chuang J H. A Simple Yet effective user controllable mesh simplification [J]. Journal of Information Science & Engineering, 2013, 29 (3) .

[10] 黄佳, 温佩芝, 李丽芳, 等. 保持细节特性的局部误差渐进网格简化算法 [J]. 计算机应用, 2016, 36 (6): 1704-1708. (Huang Jia, Wen Peizhi, Li Lifang, etal. Local error progressive mesh simplification algorithm for keeping detailed features [J]. Journal of Computer Applications, 2016, 36 (6): 1704-1708. )

[11] 王竣, 王修晖. 基于边曲率的网格模型简化算法 [J]. 中国计量学院学报, 2016, 27 (1): 73-79. (WangJun, Wang Xiuhui. Mesh simplification algorithms based on edge curvature metrics [J]. Journal of China Jiliang University, 2016, 27 (1): 73-79. )

[12] 钱勋波, 罗立宏. 一种保持特征的网格简化算法 [J]. 机电工程, 2017, 34 (10): 1224-1228. (Qian Xunbo, Luo Lihong. Method of mesh simplification based on feature preserving [J]. Mechanical & Electrical Engineering Magazine, 2017, 34 (10): 1224-1228. )

[13] 刘峻, 范豪, 孙宇, 等. 结合边折叠和局部优化的网格简化算法 [J]. 计算机应用, 2016, 36 (2): 535-540. (Liu Jun, Fan Hao, Sun Yu, etal. Mesh simplification algorithm combined with edge collapse and local optimization [J]. Journal of Computer Applications, 2016, 36 (2): 535-540. )

[14] Du Q, Faber V, Gunzburger M. Centroidal Voronoi tessellations: applications and algorithms [J]. SIAM Review, 1999, 41 (4): 637–676.




[15] Lévy B, Petitjean S, Ray N, etal. Least squares conformal maps for automatic texture atlas generation [J]. Acm Transactions on Graphics, 2006, 21 (3): 362-371.

[16] 刘泗岩, 廖文和, 刘浩. 基于内角余弦和的三角形正则度评定与网格优化 [J]. 机械科学与技术, 2007, 26 (4): 420-423. (Liu Siyan, Liao Wenhe, Liu Hao. Triangle regularity measurement based on cosine sum of inner angles and mesh optimization [J]. Mechanical Science and Technology for Aerospace Engineering, 2007, 26 (4): 420-423. )

[17] Surazhsky V, Gotsman C. Explicit surface remeshing [C]// Proc of Eurographics//ACM SIGGRAPH symposium on Geometry processing. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2003: 20-30.

[18] Cignoni P, Rocchini C, Scopigno R. Metro: Measuring error on simplified surfaces [J]. Computer Graphics Forum, 1998, 17 (2): 167-174.